WEST

Generate Collection | Print

L2: Entry 29 of 29   File: DWPI   Jun 9, 1998

DERWENT-ACC-NO: 1998-347922
DERWENT-WEEK: 199830
COPYRIGHT 2003 DERWENT INFORMATION LTD

TITLE: Concurrent operating method for multiple operating systems - involves loading
number of subdominant and operating system environments separately, independent of
first dominant operating environment

INVENTOR: LOUCKS, L K

PATENT-ASSIGNEE: INT BUSINESS MACHINES CORP (IBMC)

PRIORITY-DATA: 1993US-0023666 (February 26, 1993), 1996US-0760158 (December 3, 1996)

PATENT-FAMILY:

PUB-NO	PUB-DATE	LANGUAGE	PAGES	MAIN-IPC
US 5764984 A	June 9, 1998		008	G06F013/00

APPLICATION-DATA:

PUB-NO	APPL-DATE	APPL-NO	DESCRIPTOR
US 5764984A	February 26, 1993	1993US-0023666	Cont of
US 5764984A	December 3, 1996	1996US-0760158	

INT-CL (IPC): G06 F 13/00

ABSTRACTED-PUB-NO: US 5764984A
BASIC-ABSTRACT:

The method involves booting a first operating system as a dominant operating
environment providing physical system coordination services to all sub-dominant
operating systems. A number of sub-dominant operating system environments are loaded
separately, independent of the first dominant operating environment.

An application program is executed as an application process and interprocess
communication requests are serviced between the application process and the dominant
operating environment. Subdominant operating system environments use kernel
services.

ADVANTAGE - Reduces cost required for adapting new hardware platform. Improves
communication efficiency with hardware resources shared by multiple systems.

ABSTRACTED-PUB-NO: US 5764984A
EQUIVALENT-ABSTRACTS:

CHOSEN-DRAWING: Dwg.3/3

DERWENT-CLASS: T01
EPI-CODES: T01-F05G;

WEST

L2: Entry 11 of 29   File: USPT   Mar 11, 2003

DOCUMENT-IDENTIFIER: US 6532538 B1
TITLE: Method and system for supporting multiple operating systems on the same disk running on different computers at the same time

Abstract Text (1):
A method and system for running, on different computers at the same time, multiple operating systems from the same shared system resource is provided. This is accomplished, for example, by using persistent elemental disk reservations. Each machine reads the master boot record without reservation to determine the partition of the operating system to be booted. Each machine then makes an elemental exclusive write persistent reservation for accessing the operating system boot partition. This is followed by each machine making another elemental exclusive write persistent reservation for accessing the operating system partition itself. Each machine is assigned a different operating system partition even if they are running the same operating system. The unique reservation key for these reservations is created from at least on of a Processor ID, a Cluster ID, a Multiple Processor partition ID, a Non-Uniform Memory Access complex ID, and/or a Non-Uniform Memory Access node ID.

Brief Summary Text (5):
Loading and running an operating system (OS) is accomplished by using a bootstrap program. Normally, starting the operating system is a two step process involving a "simple" boot program that determines which operating system to load and a more complex boot program that actually loads the selected operating system. The simple boot program, usually stored in nonvolatile system RAM (NVRAM), is used to perform system resource initialization. Specifically, the simple boot program initializes registers in the Central Processing Unit (CPU) and initializes device controllers, such as controllers for the system disk and memory. The simple boot program can read and write to memory and can load from a boot block on the system disk. The boot block contains the master boot record (MBR) and is located at sector 0 of the system disk drive.

Brief Summary Text (6):
The master boot record (MBR) is loaded from the system disk and contains a partition table and some executable code. The master boot record executable code scans the partition table for a single active partition, loads the first sector from the active partition into memory, and executes this code, which is the boot code for the selected operating system. This operating system boot code loads the operating system that is being booted and starts the operating system in a defined manner.

Brief Summary Text (7):
If a hard disk contains, for example, an MS-DOS partition, a LINUX partition, a Windows NT partition, and an IBM OS/2 partition, a user can change which of these systems will be started by changing the active partition. The active partition may be set by storing this information in nonvolatile system RAM (NVRAM). Normally the last operating system installed on the system disk drive updates the NVRAM so that this operating system will be booted. But an operating system can provide a utility program that allows a different operating system to be designated as the active partition in NVRAM. This then allows the next reboot to boot to a different operating system.

Brief Summary Text (8):
Although the above method allow the user to select an operating system to be booted at system startup, this method does not allow two or more operating systems on different machines to run at the same time from the same system disk. Thus, any machine making use of a shared system disk having, for example, an OS/2 partition as

the active partition must itself run under the OS/2 operating system. This limits the versatility of the system and places restrictions on the system resources, such as the type of file system, that can be accessed. Therefore, it would be advantageous to have a method and a system for accessing a shared system resource, such as a system disk drive, so that different machines may run different operating systems and access appropriate system resources at the same time.

Brief Summary Text (10):
The present invention provides a method and system for running, on different computers at the same time, multiple operating systems from the same shared system resource. This is accomplished, for example, by using persistent elemental disk reservations. Each machine reads the master boot record without reservation to determine the partition of the operating system to be booted. Each machine then makes an elemental exclusive write persistent reservation for accessing the operating system boot partition. This is followed by each machine making another elemental exclusive write persistent reservation for accessing the operating system partition itself. Each machine is assigned a different operating system partition even if they are running the same operating system. The unique reservation key for these reservations is created from at least on of a Processor ID, a Cluster ID, a Multiple Processor partition ID, a Non-Uniform Memory Access complex ID, and/or a Non-Uniform Memory Access node ID.

Drawing Description Text (6):
FIG. 4 is a flowchart outlining an exemplary operation for updating a master boot record when a new operating system is added to the system in accordance with one embodiment of the present invention; and

Drawing Description Text (7):
FIG. 5 is a flowchart outlining an exemplary operation of a boot process in accordance with one embodiment of the present invention.

Detailed Description Text (6):
In the depicted example, distributed data processing system 100 is an Internet with network 102 representing a worldwide collection of networks and gateways that use the TCP/IP suite of protocols to communicate with one another. At the heart of the Internet is a backbone of high-speed data communication lines between major nodes or host computers, consisting of thousands of commercial, government, educational and other computer systems that route data and messages. of course, distributed data processing system 100 also may be implemented as a number of different types of networks, such as for example, an intranet, a local area network (LAN), or a wide area network (WAN). For example, on a LAN, disk drive 106 may be shared between clients 108, 110, 112, and each of these clients may have booted a different operating system resident on disk 106.

Detailed Description Text (7):
FIG. 1 is intended as an example, and not as an architectural limitation for the present invention. For example, with respect to the present invention, the "computers" may be processors in a single machine with multiple processors provided each processor has it own memory partition. In this case, multiple operating systems could be run simultaneously on the "same machine" provided each operating system is assigned its own processor and its own memory partition.

Detailed Description Text (12):
For example, data processing system 200, if optionally configured as a network computer, may not include SCSI host bus adapter 212, hard disk drive 226, tape drive 228, and CD-ROM 230, as noted by dotted line 232 in FIG. 2 denoting optional inclusion. In that case, data processing system 200 may include a network communication interface, such as LAN adapter 210, modem 222, or the like. As another example, data processing system 200 may be a stand-alone system configured to be bootable without relying on a network communication interface, whether or not data processing system 200 comprises a network communication interface. The depicted example in FIG. 2 and above-described examples are not meant to imply architectural limitations. For example, the present invention may be implemented in multiple processor systems, such as a non-uniform memory access (NUMA) computer.

Detailed Description Text (14):
Disk drive 300 is accessed by a plurality of machines, which, for simplicity, are shown with reservation IDs 1 through 7. Each computer running an operating system on disk 300 is assigned a separate partition for its operating system and uses its

reservation ID as a key to reserve this partition. In the depicted example, computer 302 is currently accessing master boot record 304 at the start of the boot process. Computer 306 has already read the master boot record and determined it is to boot the OS/2 operating system. It is now executing OS/2 boot partition 308. Computer 310 is currently running the Windows NT operating system, a product of Microsoft Corporation. Its reservation ID (# 3) is used as a key to reserve an elemental persistent reservation for Windows NT partition 312 on disk 300. Computer 314 is running Linux and has reserved Linux partition 316 on disk 300 using its reservation ID (# 4) as a key. Computer 318 is running MS-DOS and has reserved MS-DOS partition 320 on disk 300 using its reservation ID (# 5) as a key. Computer 322 is running Linux, but notice that its Linux partition 324 on disk 300 is different than Linux partition 316 reserved by computer 308. Computer 326 is running OS/2 and has reserved OS/2 partition 328 on disk 300 using its reservation ID (# 7) as a key. Thus, by using different reservation IDs and different partitions on a shared disk drive, different computers can simultaneously run different operating systems from the shared disk drive.

Detailed Description Text (15):
In FIG. 3 reservation IDs are shown as simple integers. As one of ordinary skill in the art will appreciate, actual values may be more complex. In addition, the illustration of partitions on disk drive 300 in FIG. 3 does not represent actual relative amounts of physical disk space. For example, the master boot record may take only one sector of the disk while an operating system partition may take tens of thousands of sectors.

Detailed Description Text (16):
With reference now to FIG. 4, a flowchart outlines an exemplary operation for updating a master boot record when a new operating system is added to the system in accordance with one embodiment of the present invention. For example, the partition table in the master boot record must be changed each time an operating system partition, such as partitions 312, 316, 320, 324, and 328 in FIG. 3, is added to disk drive 300. When an operating system is installed, a partition on the system resource, such as disk drive 300, is reserved and the partition information is added to master boot record 304. It is important that two operating systems do not attempt to modify the master boot record at the same time during the installation process otherwise the partition table in the master boot record will not contain the correct information for the multiple operating systems.

Detailed Description Text (17):
The operating system or a utility program acting on behalf of the operating system makes an elemental exclusive write persistent reservation on the master boot record area (step 400). If the reservation does not succeed (step 402: No), then the modification to the master boot record fails (step 404) and an error signal is sent to the operating system installation program (step 406). If the reservation on the operating system boot partition succeeds (step 402: Yes), the operating system or utility program adds the boot partition information to the master boot record (step 408). The operating system or utility program then releases the elemental exclusive write persistent reservation on the master boot record area (step 410) and the modification of the master boot record ends.

Detailed Description Text (19):
With reference now to FIG. 5, a flowchart outlines an exemplary operation of the boot process in accordance with one embodiment of the present invention. It is assumed at this point that two or more operating systems have been installed successfully on the same system resource. The machines in FIG. 3 are shown at various stages of booting an operating system. Computer 302 is reading master boot record 304, computer 306 is reading OS/2 boot partition 308, and computers 310, 314, 318, 322, and 326 have completed the boot process.

Detailed Description Text (20):
Initially, the boot code reads the system resource without reservation (step 500). Then, the boot code registers a persistent reservation key with the system resource (step 502). The boot code reads the master boot record (MBR) and determines the partition for the operating system to be booted based on the active operating system partition information stored in NVRAM (step 504). The boot code then makes an elemental exclusive write persistent reservation on the operating system boot partition (step 506). If the reservation did not succeed (step 508: No), then the boot fails (step 510).

Detailed Description Text (21):
If the reservation on the operating system boot partition succeeds (step 508: Yes),
then the code in the operating system boot partition is executed (step 512). The
operating system boot partition makes an elemental exclusive write persistent
reservation on the operating system partition (step 514). If the reservation did not
succeed (step 516: No), then the boot fails (step 518). If the reservation on the
operating system boot partition succeeds (step 516: Yes), then the code in the
operating system partition is executed (step 520).

Detailed Description Text (22):
In order to keep multiple operating systems from interfering with each other, each
operating system has a unique reservation key. In addition, the integrity of the
operating system boot process is preserved by having the operating system boot code
place an elemental persistent reservation on the operating system partition in an
environment where there may be boot processes on other machines booting the same
operating system from the same disk drive. It is important to note that the file
system organization for the various operating system partitions need not be
compatible. All that is required is that each of the boot programs can read and
interpret the information in the master boot record correctly.

Detailed Description Text (23):
In summary, in the prior art, two or more operating systems could be booted from the
same disk but not run simultaneously, even when these operating systems are running
on different machines. By using persistent elemental disk reservations in the manner
described above as presented in this invention, it is possible to have multiple
operating systems running on different machines from the same system resource at the
same time.

Detailed Description Text (24):
Although the above description of the invention assumed a plurality of computers
sharing a system disk drive, the invention is not limited to such as embodiment. In
particular, the method developed in this invention could be extended to a machine
with multiple processors where each processor has its own memory partition and
operating under its own operating system. In this case, multiple operating systems
could be run simultaneously on the "same machine" provided each operating system is
assigned its own processor and its own memory partition. As those skilled in the art
will appreciate, a similar setup would be possible for NUMA (Non-Uniform Memory
Access) complexes.

CLAIMS:

1. A method for booting a plurality of operating systems on a plurality of
processors from a shared system resource, the method comprising: reading a master
boot record from the shared system resource by a processor from the plurality of
processors to determine the partition of an operating system from the plurality of
operating systems to be used; reserving a portion of the shared system resource
using a unique reservation key by the processor for accessing an operating system
boot partition; and reserving a portion of the shared system resource using the
unique reservation key by the processor for accessing an operating system partition.

3. The method of claim 1, wherein the master boot record is initially read by the
processor from the shared system resource without reservation.

4. The method of claim 1, wherein the unique reservation key for reserving a portion
of the shared system resource for the operating system boot partition and the
operating system partition is created from at least one of the following: a
Processor ID, a Cluster ID, a Multiple Processor partition ID, a Non-Uniform Memory
Access complex ID, and a Non-Uniform Memory Access node ID.

6. A method in a distributed data processing system for executing a plurality of
operating systems on a plurality of processors in the distributed data processing
system from a shared system resource, the method comprising the processor
implemented steps of: reading a master boot record from the shared disk by each of
the plurality of processors to identify operating systems for the plurality of
processors; reserving, by each of the plurality of processors, a portion of the
shared system resource using a reservation key unique to each of the plurality of
processors, wherein the portion is used to access an operating system boot
partition; and reserving, by each of the plurality of processors, a portion of the

shared system resource using the reservation key to access an <u>operating system partition, wherein the plurality of processors concurrently execute the plurality of operating systems</u> from the shared system resource.

7. A method in a data processing system for installing a plurality of operating systems from a plurality of processors on a shared system resource, the method comprising the data processing system implemented steps of: a processor from the plurality of processors installing an operating system from the plurality of operating systems reserving exclusive access to a master <u>boot</u> record on the shared system resource; the processor adding the operating system <u>boot</u> information to a partition table in the master <u>boot</u> record; and the processor releasing the exclusive access to the master <u>boot</u> record.

8. The method of claim 7, wherein the reservation of exclusive access to the master <u>boot</u> record of the shared system resource is accomplished by an elemental exclusive <u>write</u> persistent reservation on the portion of the shared system resource that contains the master <u>boot</u> record.

9. The method of claim 7, wherein an error message is sent to an operating system install program if the elemental exclusive write persistent reservation on the master <u>boot</u> record fails.

10. A distributed data processing system for <u>booting</u> a plurality of operating systems on a plurality of processors from a <u>shared</u> system resource, the data processing system comprising: determining means for a processor from the plurality of processors reading a master <u>boot</u> record from the shared system resource to determine an operating system from the plurality of operating systems to be <u>booted</u>; reserving means for the processor reserving a portion of the shared system resource for access to an operating system <u>boot</u> partition using a unique reservation key; and reserving means for the processor reserving a portion of the shared system resource for access to an operating system partition using the unique reservation key.

12. The system of claim 10, wherein the master <u>boot</u> record is initially read by the processor from the shared system resource without reservation.

13. The system of claim 10, wherein the unique reservation key for reserving a portion of the shared system resource for the operating system <u>boot</u> partition and the operating system partition on the shared system resource is created from at least one of the following: a Processor ID, a Cluster ID, Multiple Processor partition ID, a Non-Uniform Memory Access complex ID, and a Non-Uniform Memory Access node ID.

14. A distributed data processing system for executing a plurality of operating systems on a plurality of processors in the distributed data processing system from a shared system resource, the distributed data processing system comprising: reading means for reading a master <u>boot</u> record from the shared disk by each of the plurality of processors to identify operating systems for the plurality of processors; reserving means for reserving, by each of the plurality of processors, a portion of the shared system resource using a reservation key unique to each of the plurality of processors, wherein the portion is used to access an operating system <u>boot</u> partition; and reserving means for reserving, by each of the plurality of processors, a portion of the shared system resource using the reservation key to access an <u>operating system partition, wherein the plurality of processors concurrently execute the plurality of operating system</u> from the shared system resource.

15. A computer program product on a computer-readable medium for use in a network workstation for <u>booting</u> a plurality of operating systems on a plurality of processors from a shared system resource, the computer program product comprising the data processing system implemented steps of: instructions for a processor from the plurality of processors reading a master <u>boot</u> record from the shared system resource to determine an operating system from the plurality of operating systems to be <u>booted</u>; instructions for the processor reserving a portion of the shared system resource for access to an operating system <u>boot</u> partition using a unique reservation key; and instructions for the processor reserving a portion of the shared system resource for access to an operating system partition using the unique reservation key.

WEST

Generate Collection   Print

L2: Entry 3 of 29   File: PGPB   Jun 19, 2003

DOCUMENT-IDENTIFIER: US 20030115443 A1
TITLE: Multi-O/S system and pre-O/S boot technique for partitioning resources and loading multiple operating systems thereon

Abstract Paragraph (1):
A technique is provided for loading a multiple-O/S platform on a computing device, such as a personal computer or server. The technique supports multiple operating systems by partitioning the computing device in an initialization or pre-boot phase, in which system resources are detected, initialized and tabulated. A desired O/S of the multiple operating systems is then loaded on each partitioned set of the system resources. The multiple operating systems may then operate simultaneously and independently on the computing device.

Summary of Invention Paragraph (5):
[0003] Accordingly, a technique is needed for consolidating the desired applications onto fewer computing devices, such as servers. More particularly, there is a need for a multi-platform operating system to facilitate the foregoing application consolidation. It would be particularly advantageous to provide a pre-boot partitioning technique, which could partition the resources of the computing device for a plurality of operating systems.

Summary of Invention Paragraph (7):
[0004] A technique is provided for loading a multiple-O/S platform on a computing device, such as a personal computer or server. The technique supports multiple operating systems by partitioning the computing device in an initialization or pre-boot phase, in which system resources are detected, initialized and tabulated. A desired O/S of the multiple operating systems is then loaded on each partitioned set of the system resources. The multiple operating systems may then operate simultaneously and independently on the computing device.

Summary of Invention Paragraph (9):
[0006] In another aspect, the present technique provides a system for booting a computing device. The system comprises a resource tabulator module, a resource divider module, and an operating system loader module. The resource tabulator module is configured to organize data on system resources for the computing device. The resource divider module is configured to create multiple resource sets for the computing device. The operating system loader module is configured to load a desired operating system on each of the multiple resource sets.

Detail Description Paragraph (2):
[0014] As described in detail below, the present technique provides a unique system 10 for operating a plurality of platforms, or operating systems, on a computing device. The present technique may be utilized in desktop computers, portable computers (e.g., laptops, notebooks, palmtops, etc.), servers, workstations, and various other electronics and computing devices. In particular, the present technique is useful for server applications, which often require various operating systems to run the desired applications. The present technique advantageously loads multiple operating systems or platforms on the desired computing device in a pre-boot stage (e.g., a ROM-based or BIOS environment), such that the multiple operating systems are all loaded onto the computing device at the same level without an underlying operating system (e.g., Windows, LINUX, or a MAC O/S). Accordingly, the system 10 interfaces with the resources of the computing device, partitions the resources, and loads the multiple operating systems on the partitions for simultaneous and independent operation on the computing device.

Detail Description Paragraph (6):
[0018] As mentioned above, the resource interface and partition system 14 utilizes a variety of hardware and software to interface with, and partition, the foregoing resources 15 into distinct resource sets, each of which supports an independent platform on the computing device 12. The system 10 performs these functions in a pre-boot or initialization phase, such as in a ROM-based or BIOS environment of the computing device 12. Accordingly, the resource interface and partition system 14 may be incorporated into one or more memory modules, such as ROM or Flash memory modules, which run during the pre-boot phase of the system 10. For example, the resource interface and partition system 14 may comprise a resource identifying module 68, a resource cataloguing module 70, a resource allocating module 72, and any other desired modules to facilitate division of the resources 15 (e.g., an extensive firmware interface) prior to an O/S boot on the computing device 12.

Detail Description Paragraph (8):
[0020] The resource allocating module 72 then proceeds to allocate the resources 15 into a plurality of resource sets, which may share some of the resources between the resource sets. The resource allocating module 72 may comprise a resource divider module, a resource sharing module, a resource cloning module, and various other modules to facilitate exclusive and shared access and control of the resources 15. For example, the resource divider module can include a memory partitioning module, a processor partitioning module, and other partitioning modules for creating independent computing clusters from the resources 15. The resource sharing module provides shared control of certain necessary or critical resources 15, such as multi-device driver modules (e.g., a multi-display driver), SAL services, PAL services, input/output services, interrupt services, boot services, runtime services, and various other system and processor services (e.g., modules 18). The resource cloning module also facilitates sharing by generating and distributing instances of at least part of the resources 15, such as boot service modules, runtime service modules, and O/S loader modules.

Detail Description Paragraph (9):
[0021] For example, the system 10 may allocate the resources 15 into a plurality of resource sets, such as resource sets 74 and 76 (i.e., Resource Sets #1 and #2), which may comprise a variety of shared resources 78. Each of the resource sets 74 and 76 can include a portion of the processors 16 (e.g., a CPU cluster), a portion of the memory 34, a portion of the input/output ports 36 and corresponding input devices 38 and output devices 40, one or more of the disk drives 42, and various other portions of the resources 15. The resource sets 74 and 76 also may include all or part of the system and processor control modules 18, such as boot services and runtime environment descriptors (e.g., ACPI and SST tables). The system 10 also may utilize USB architecture to obviate the need for common device drivers, such as a common keyboard driver. As discussed above, the shared resources 78 may comprise a variety of hardware and software, such as the input/output modules 28, the interrupt modules 32, the SAL/PAL modules 22, a dual screen video driver, and various other resources.

Detail Description Paragraph (13):
[0025] As described above, and illustrated by process 88 of FIG. 2, the multi-platform system 10 is configured and operated by partitioning the resources 15 during a pre-boot phase of the computing device 12, and then loading multiple operating systems on the respective resource sets. For example, the process 88 may proceed by powering the computing device (block 90) into a pre-boot or initialization phase, wherein the process 88 detects and initializes resources 15 of the computing device (block 92). As described above, the process 88 may configure the resources 15 automatically or manually via a ROM-BIOS module or any other suitable configuration module, which may be disposed on ROM, RAM, EEPROM, Flash Memory, a floppy disk, a remote network drive, or any other suitable storage device. For example, the pre-boot or initialization phase may configure the basic hardware devices of the computing device. This generally occurs prior to booting the primary operating system (e.g., Windows, LINUX, MAC O/S, etc.), which supports the desired applications software for the computing device. During this pre-boot phase, the process 88 may execute the input/output modules 28, the device driver modules 30, and various other system and processor control modules 18. Accordingly, the process 88 detects the hardware devices 94 and executes the input/output and device drivers 96 to facilitate interaction with the resources 15 during the subsequent stages of configuring and loading multiple operating systems.

Detail Description Paragraph (15):

[0027] The foregoing resource sets 102, 104 and 106 then support a plurality of independent application platforms or operating systems, which are simultaneously and independently operable on the computing device 12. For example, the process 88 may boot or load multiple operating systems, such as operating systems 110, 112 and 114 (i.e., O/S's #1, O/S #2 and O/S #N), on the multiple sets of resources (block 108). The operating systems 110, 112 and 114 may comprise identical or different operating systems, such as Windows, LINUX, and a MAC O/S, each of which may be utilized independently on the computing device 12 via the resource sets 102, 104 and 106 (block 116). The process 88 can then load and run the desired applications, such as software applications 118, 120 and 122 (i.e., APPS #1, #2 and #N), on each of the operating systems 110, 112 and 114 (i.e., O/S's #1, #2 and #N), respectively (block 124). As the user utilizes the operating systems 110, 112 and 114 and the corresponding applications 118, 120 and 122, the respective resource sets 102, 104 and 106 provide exclusive and shared access, use and control of the hardware and software components within those resource sets 102, 104 and 106 (Block 124).

Detail Description Paragraph (16):
[0028] FIG. 3 illustrates an exemplary embodiment of the system 10, wherein the resources 15 are partitioned for simultaneously running multiple operating systems. As illustrated, the system 10 utilizes the interface 14 to partition the resources 15 and initiate loading of multiple operating systems. In this exemplary embodiment, the interface 14 embodies an extensible firmware interface ("EFI") for logical partitioning by using information obtained from the BIOS, ACPI tables, SST tables, MP tables, etc. The system also provides a user interface 126 for interacting with the extensible firmware interface 14 to facilitate partitioning of the resources 15. In operation, the interface 14 spawns multiple instances of native EFI boot services and run-time environment descriptors (e.g., a CPI, SST, and MP tables), such as illustrated by resource groups 128 and 130, which correspond to the multiple partitions. As illustrated, the resource groups 128 and 130 correspond to partitions "0" and "N," which may be any desired number of partitions and operating systems for the computing device 12. In FIG. 3, only two such partitions are provided for illustration purposes.

Detail Description Paragraph (17):
[0029] Resource group 128 may comprise a variety of system and processor resources, such as boot services 132, run-time services 134, ACPI(0) tables 136, SST(0) tables 138, CPU clusters(0) 140, SMBIOS(0) data 142, MP tables(0) 144, system abstraction layer SAL(0) 146, and processor abstraction layer PAL(0) 148. Similarly, resource group 130 may comprise a variety of system and processor resources, such as boot services 150, run-time services 152, ACPI(N) tables 154 to, SST(N) tables 156, CPU clusters(N) 158, SMBIOS(N) data 160, MP tables(N) 162, system abstraction layer SAL(N) 164, and processor abstraction layer PAL(N) 166. The extensible firmware interface 14 creates the foregoing resource groups 128 and 130 independent of the particular BIOS disposed on the computing device 12.

Detail Description Paragraph (18):
[0030] The extensible firmware interface 14 then proceeds to initiate boot loaders for each of the desired operating system, which may be loaded and simultaneously operated on the computing device 12. Accordingly, an OS boot loader 168 is initiated for partition "0," while an OS boot loader 170 is initiated for partition "N." These OS boot loaders 168 and 170 load desired operating system, such as Windows, Linux, MacOS or any other suitable operating system, onto the respective partitions "0" and "N." The respective operating systems may be the same or different, as desired for particular applications.

CLAIMS:

9. A method for simultaneously supporting a plurality of independent operating systems on a computing device, comprising: cataloguing resources of the computing devices prior to O/S booting for the computing device; dividing the resources into multiple subsets prior to O/S booting; and loading the plurality of independent operating systems, at least one O/S being loaded on each resource set of the multiple subsets.

14. A system for booting a computing device, comprising: a resource tabulator module configured to organize data on system resources for the computing device; a resource divider module configured to create multiple resource sets for the computing device; and an operating system loader module configured to load a desired operating system on each of the multiple resource sets.

15. The system of claim 14, wherein the resource tabulator module and the resource divider module are disposed in a pre-boot environment.

17. The system of claim 14, wherein the pre-boot environment comprises hardware detection modules for the system resources.

18. The system of claim 14, wherein the pre-boot environment comprises hardware driver modules for the system resources.

꺼 ⁄꾸꺼

WEST

☐ | Generate Collection | | Print |

DERWENT-ACC-NO: 2002-617466
DERWENT-WEEK: 200266
COPYRIGHT 2003 DERWENT INFORMATION LTD

TITLE: Assistant operating system booting system in computer, has system boot card
with data storage unit for storing programs in assistants operating system

INVENTOR: HUNG-JU, S; LIN, Y

PATENT-ASSIGNEE: HUNG-JU S (HUNGI), LIN Y (LINYI)

PRIORITY-DATA: 2000TW-0126853 (December 15, 2000)

PATENT-FAMILY:

PUB-NO	PUB-DATE	LANGUAGE	PAGES	MAIN-IPC
US 20020078339 A1	June 20, 2002		005	G06F015/177

APPLICATION-DATA:

PUB-NO	APPL-DATE	APPL-NO	DESCRIPTOR
US20020078339A1	January 17, 2001	2001US-0761942	

INT-CL (IPC): G06 F 9/00; G06 F 15/177

ABSTRACTED-PUB-NO: US20020078339A
BASIC-ABSTRACT:

NOVELTY - An interface control module in a system boot card is connected to an
input/output interface in a host computer having a main operating system. A data
storage unit in the system boot card stores the programs in the assistant operating
system so that capability of the host computer is enhanced after downloading the
assistant operating system into the computer and executing main and assistant
operating systems simultaneously by a processor in the computer.

DETAILED DESCRIPTION - An INDEPENDENT CLAIM is included for assistant operating
system booting method.

USE - For booting assistant operating system in computer.

ADVANTAGE - The assistant operating system is detached from the main operating
system, hence the amount of storage space in the host system for holding the
operating system is reduced and the time for the host system in normal operation to
initiate the operating system is shortened and the size, complexity and downloading
time of the main operating system are reduced. The re-programmable or multiple card
insertion type of system boot card provides a variety of different functions in the
assistant operating on demand.

DESCRIPTION OF DRAWING(S) - The figure shows the flowchart illustrating the
assistant operating system booting procedure.

ABSTRACTED-PUB-NO: US20020078339A
EQUIVALENT-ABSTRACTS: